Assignment 1.1: Web Service (WDSL/SOAP)

Objectives:

In the lectures you have been exposed to a number of concepts related to Web services, to help you grasp these concepts, we have defined a simple assignment in which you will have to develop a simple web service namely " the calculator web service". You will develop the service first as WSDL/SOAP Web Service then as RESTFUL service. By developing simple service using two different methods, you will learn the two different ways of developing web services and hopefully will be able to choose the most appropriate solution for your specific problem.

Background:

There are two major approaches to develop WDSL/SOAP services: (1) top-down and (2) Bottom-up:

- **Bottom Up Approach of Web services Development**

  The bottom up approach (Code First) is used where the developer starts with the business logic, which is the code, and then develops and deploys the code as a Web service. See [1] for further details about this approach. Follow the steps described in the Tutorial to learn the Bottom Up Approach of Web services Development

- **Top Down Approach of Web services Development.**

  The top down approach (Contract First) takes the contract as the primary artifact. The "contract" for a web service is the WSDL document. See [2] for further details about this approach. Follow the steps described in the Tutorial to learn the Bottom Up Approach of Web services Development

Assignment:

1. **Prepare** the Lab by following the tutorials [1] and [2]. We advise you to follow the tutorials at home and download and configure all the software needed for the lab. Web services can be developed in both Windows and Linux based environments, you only need to install a service container like Tomcat or glassfish. If you're using an IDE like Eclipse or NEtBeans it is likely that the IDE has already all the packages needed to develop web services [5][4].

2. **Write a Calculator** Web service using the two methods you have learned (Bottom-up or Top-Down). Write the respective clients to call your service in each case.

    - Implement a web service using Java API for XML Web Services (JAX-WS) with the following interface:
        - method add for addition
        - method sub for substraction
        - method mul for multiplication
        - method div for division
    - Calculator should operate on real numbers
    - Web service should communicate using SOAP protocol

3. **Propose a solution** to make the Calculator stateful (do not implement just describe the solution: text + diagram)

4. **How to get your assignment GRADED:**
    - Show a working prototype to the Lab assistant
    - **Submit via Blackboard a** tar file containing
        - All the needed files to deploy your service including
        - Readme file to explain how to deploy and run the client.
        - A short report describing your design and implementation of the calculator service (at most 1 A4 page)
        - The answer to question 3
        - The tar file should named: <group-number>_web_service_1.tar

References:

1. http://docs.oracle.com/javaee/5/tutorial/doc/bnayn.html
2. http://docs.oracle.com/cd/E14571_01/web.1111/e13735/handlers.htm#i222633
3. http://sewiki.iai.uni-bonn.de/teaching/lectures/ise/2008/assignment_i
4. http://sewiki.iai.uni-bonn.de/teaching/lectures/ise/2008/assignment_ii
5. http://sewiki.iai.uni-bonn.de/teaching/lectures/ise/2008/assignment_iii
6. develop web service in Netbean http://www.oracle.com/technetwork/java/deploy-nb-141966.html
7. Eclipse WTP Tutorials- Crating a Bottom Up Web service via Apache Axis2 http://www.eclipse.org/webtools/community/tutorials/BottomUpAxis2WebService/bu_tutorial.html

Assignment 1.2: RESTFUL service

Background:

REST (Representational State Transfer) has emerged in the last few years as a predominant Web service design model. It is an alternative to SOAP and WSDL Web services (you have developed in the first part of the assignment). REST is now the mainstream approach in developing Web 2.0 service.

Assignment:

1. **Prepare for the assignment**: There are plenty of good online tutorials to help you develop and deploy your first RESTful service [2-6]. The most popular framework to develop RESTful service is "Jersey" (Jersey is the open source, JAX-RS (JSR 311) Reference Implementation for building RESTful Web services [1]). Consider using Ruby on Rails [14], Sinatra.rb [15] or Python microframework – flask [16.

2. **Write a REST Calculator** service.

- Calculator can operate on equations expressed in conventional notation (e.g. (1+2)*3), polish notation (PN), (e.g. * 3 + 1 2) or reversed polish notation (RPN) (e.g. 1 2 + 3 *),
- For full amount of points, equations are suppose to be checked if they are syntactically correct (we suggest to use reversed polish notation),
- Calculator is suppose to support four operators: +, -, *, : (addition, subtraction, multiplication and division) and numbers in a format `(\d{1,5}\.?\d{0,5})|(\d{0,5}\.?\d{1,5})` – up to 5 digits before and after a decimal mark, (e.g. `1`, `2.`, `.4`, `231.434`)
- Numerical precision of calculations will not influence the final grade
- For stateless calculator implement the following interface
- A client that can talk to this interface is available on github [13]

Path and method	Parameters	Return value (HTTP code, value)
`/calc/:equation` - `GET`	`:equation` – represents equation, for PN and RPN elements are separated with '&' symbol, e.g. `1&2&+&3&*`	200, value 400, "division by zero" 400, "syntax error"

3. **Consider implementing stateful calculator** (on github [13] there is a client that implements described protocol and can check your service). You can also propose other solution to make the REST Calculator stateful (do not implement just describe the solution: text + diagram)

- For stateful calculator implement the following interface
  - for items

	Parameters	Return value (HTTP code, value)
/calc2/:id - GET	:id – equation id	200, value 404, -
/calc2/:id - PUT	:id – equation id :equation – as in a stateless calculator but instead of number can consist ACC string which is replaced with current values of the equation	200, - 400, "division by zero" 400, "syntax error" 404, -
/calc2/:id – DELETE	:id – equation id	204, - 404, -

  - for collection

	Parameters	Return value (HTTP code, values)
/calc2 - GET		200, :keys
/calc2 - POST	:equation – as in a stateless calculator	201, :id 400, "division by zero" 400, "syntax error"
/calc2 - DELETE		204, -

4. **How to get your assignment GRADED:**
   - Show a working prototype to the Lab assistant
   - **Submit via Blackboard a** tar file containing
     - All the needed files to deploy your service including
     - Readme file to explain how to deploy and run the client.
     - A short report describing your design and implementation of the calculator service
     - The answer to question 3
     - The tar file should named: <group-number>_web_service_1.2.tar

Other Alternative:

- **RESTEasy** is another framework to develop RESTful services. **RESTEasy** is a JBoss project that provides various frameworks to help you build

RESTful Web Services and RESTful Java applications [7]. You can also do the exercise using RESTEasy if you are familiar with it or you are already JBoss developer (see [8] Tutorial for developing web Services using RESTEasy).

- Web service do not have to be developed in Java, you can also develop Web services in other languages like Python, or Ruby. If you like to try it here are couple of references [10-12].

References:

1. Jersay web page: https://jersey.java.net
2. How to create a simple Restful Web Service using Jersey JAX RS API http://theopentutorials.com/examples/java-ee/jax-rs/create-a-simple-restful-web-service-using-jersey-jax-rs/
3. REST with Java (JAX-RS) using Jersey – Tutorial http://www.vogella.com/articles/REST/article.html
4. http://www.bhaveshthaker.com/13/introduction-developing-implementing-restful-web-services-in-java/
5. Build a RESTful Web service using Jersey and Apache Tomcat http://www.ibm.com/developerworks/library/wa-aj-tomcat/
6. RESTful Java Client With Jersey Client http://www.mkyong.com/webservices/jax-rs/restful-java-client-with-jersey-client/
7. http://www.jboss.org/resteasy.
8. RESTful Java Client With RESTEasy Client Framework http://www.mkyong.com/webservices/jax-rs/restful-java-client-with-resteasy-client-framework/
9. RESTful web services with Python. The easy way http://blog.webspecies.co.uk/2011-06-15/restful-web-services-with-python-the-easy-way.html
10. http://rest.elkstein.org/2008/02/using-rest-in-python.html
11. REST services in Python https://portal.futuregrid.org/rest-services-python
12. Beginners guide to creating a REST API (Ruby) http://www.andrewhavens.com/posts/20/beginners-guide-to-creating-a-rest-api
13. https://github.com/mikolajb/soa-cloud-course
14. http://rubyonrails.org/
15. http://www.sinatrarb.com/
16. http://flask.pocoo.org/

IMPORTANT NOTE:

- Lab Teachers and the rest of the team coordinating the course will support you in doing your assignment during Lab Session or by reacting to your emails.
- Lab Teachers have extensive experience in developing in Java, Ruby and Python will be able to provide you with High quality support if you develop in these two languages.
- Lab Teachers will provide you with **best Effort support** if you decide to develop in another language or use another environment than the one suggested in the assignment.
- All environments and methods listed in the "**Other Alternative" Section** are subject to **best Effort support.**